Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)

Факультет «Робототехника и комплексная автоматизация»

Кафедра «Системы автоматизированного проектирования»

**Отчет по лабораторной работе**

По курсу «Объектно-ориентированное программирование»

Выполнил: Студент Петраков С.А.

Группа РК6-26Б

Проверил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2020 г.

**Вариант ПЯВУ E16**

**Задание:**

Разработать объектно-ориентированную программу для закраски окна алфавитно-цифровой консоли поочередно во все цвета фона в порядке роста их escape-кодов. Закраска должна производиться в визуально-различимом темпе по 2 рядов окна консоли от центра к краям. Чередование фонов должно продолжаться до завершения программы при любом консольном вводе или по сигналу ^C. При этом должен быть восстановлен исходный фон с очисткой окна консоли. Программная реализация должна предусматривать разработку манипуляторов потока стандартного вывода для строк escape-последовательностей. Их программный код вместе с классом escape-потока, перегрузкой оператора вывода для его объектов и функцией контроля консольного ввода должен быть сосредоточен в консольном пространстве имен. Цикл перекраски должна кодировать основная функция программы.

**Алгоритм:**

Программа закрашивает окно консоли поочередно во все цвета фона в порядке роста их escape-кодов от центра к раям по 2 ряда. При нажатии любой клавиши программа перестаёт работать, очищая при этом экран. С помощью escape последовательностей переносим курсор каждый раз по полю консоли и красим в соответствующий цвет.

**Текст программы:**

#include <iostream>

#include <sstream>

#include <string>

#include <stdio.h>

#include <cstdlib>

#include <sys/ioctl.h>

#include <unistd.h>

#include <signal.h>

#include <termios.h>

//=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

// infinity loop control

bool end = 0;

//ctrl+c func

void interrupt(int signo)

{

end = signo;

return;

}

//return n>0 if key is pressed

int keyPress()

{

int n = 0;

char buf[512];

struct termios t[2];

tcgetattr(0, &t[0]);

tcgetattr(0, &t[1]);

t[0].c\_lflag &= ~(ICANON | ECHO);

t[0].c\_cc[VMIN] = 0;

t[0].c\_cc[VTIME] = 0;

tcsetattr(0, TCSANOW, &t[0]);

n = read(0, buf, 512);

tcsetattr(0, TCSAFLUSH, &t[1]);

return n;

}

//=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

namespace testNamespace

{

//return max symbols in lines

int getMaxCol()

{

struct winsize w;

ioctl(0, TIOCGWINSZ, &w);

return w.ws\_col;

}

//return max lines

int getMaxRow()

{

struct winsize w;

ioctl(0, TIOCGWINSZ, &w);

return w.ws\_row;

}

//escape-manipulators for clear screen

std::ostream& CLR(std::ostream& s)

{

return s << "\033[m\x1B[2J\033[1;1H" << std::flush;

}

//Class for storing ESC-codes

class escapeCode

{

private:

std::string escape;

public:

escapeCode(std::string e)

{

escape = e;

}

friend std::ostream& operator << (std::ostream&, escapeCode);

};

//Output ESC-code

std::ostream& operator << (std::ostream& s, escapeCode e)

{

s << e.escape << std::flush;

return s;

}

//place curson on row x, and col y

escapeCode GOTOXY(int x, int y)

{

std::ostringstream s;

s << "\033[" << y << ";" << x << "H";

return escapeCode(s.str());

}

//Set background color in x

/\*

Black 0

Red 1

Green 2

Yellow 3

Blue 4

Magenta 5

Cyan 6

White 7

\*/

escapeCode BACKGRONDCOLOR(int x)

{

std::ostringstream s;

s << "\033[4" << x << "m";

return escapeCode(s.str());

}

}

int main()

{

//=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

//Init module

signal(SIGINT, interrupt);

int colL, colR, currLine = 1;

int color = 0;

int maxCol = testNamespace::getMaxCol();

int maxRow = testNamespace::getMaxRow();

if (maxCol % 2 != 0)

colL = colR = maxCol / 2;

else

{

colL = maxCol / 2;

colR = colL + 1;

}

//=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

//Main module

std::cout << testNamespace::CLR;

while (end == 0)

{

int i = colL;

int j = colR;

//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

//One line

while (i >= 1)

{

color = color % 8;

std::cout << testNamespace::BACKGRONDCOLOR(color);

std::cout << testNamespace::GOTOXY(i, currLine) << ' ' << testNamespace::GOTOXY(i, currLine + 1) << ' ';

std::cout << testNamespace::GOTOXY(j, currLine) << ' ' << testNamespace::GOTOXY(j, currLine + 1) << ' ';

usleep(1e4);

color++;

i--;

j++;

}

//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

//Goto Next line

currLine += 2;

if ((currLine + 1) >= maxRow)

{

currLine = 1;

std::cout << testNamespace::CLR;

}

//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

if (keyPress())

break;

}

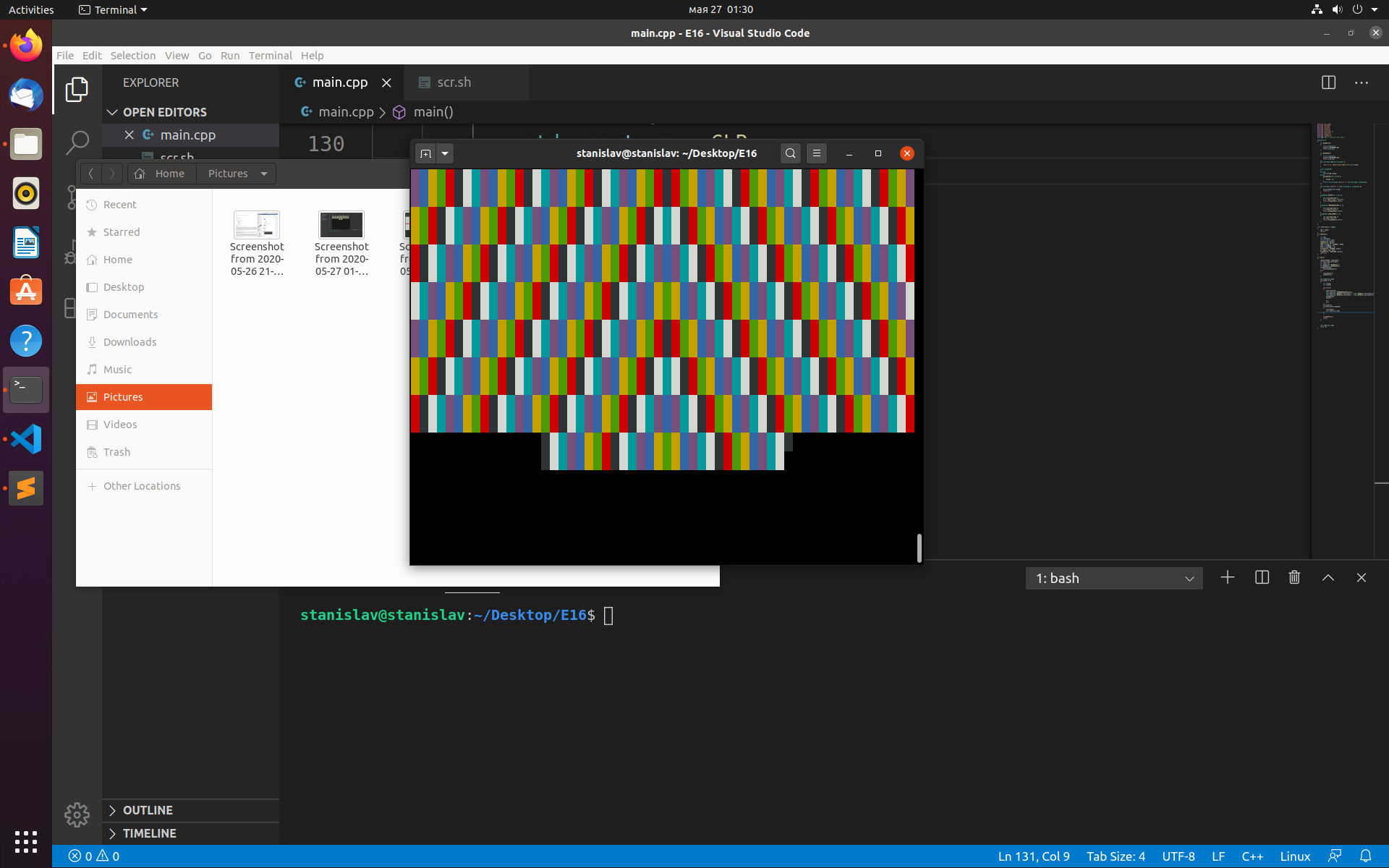
//=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

std::cout << testNamespace::CLR;

return 0;

}

**Тест:**

****

**Список использованной литературы:**

* Волосатова Т.М., Родионов С.В. Лекции по курсу «Объектно-ориентированное программирование»
* bigor.bmstu.ru